XX

Twentieth Lecture:
AUTOMATA & FORMAL LANGUAGES

Saturday, 19th November 2022

RECAP We encoded

INSTRUCTIONS
PROGRAM LINES
PROGRAMMES
REGISTER MACHINES
CONFIGURATIONS

by functions **code** in such a way that the operations that constitute the actions of the RM are performed by RM:

Operations performed & questions answered by register machines

1. Is w a code for a ...
   1. number / state?
   2. instruction?
   3. program line?
   4. register machine?
   5. sequence of words?
   6. configuration?
2. If w is a code for a register machine, which instruction does it have for state q?
3. If w is a code for a sequence of words, how long is it?
4. If w is a code for a configuration, which state is it in?
5. Given $\mathrm{code}(C)$ and $\mathrm{code}(I)$, apply I to C and output the code of the resulting configuration.
6. Given $\mathrm{code}(C)$ and $\mathrm{code}(M)$, output the code of the configuration that is the transformation of C by M.

From Lecture XIX

Lemma 4.23 The function

$$h: w,v,v \longmapsto \begin{cases} \mathrm{code}(C(M,\vec{w},\#v)) & \text{if there are } M,\vec{w} \text{ s.t. } w=\mathrm{code}(M),\ v=\mathrm{code}(\vec{w}) \\ \uparrow & \text{o/w} \end{cases}$$

is computable.

Corollary 4.24 The function

$$t_{M,k}(\vec{w},v) := \begin{cases} q & M \text{ has halted before time } \#v \text{ on input } \vec{w} \\ \varepsilon & \text{o/w} \end{cases}$$

called the truncated computation is computable.

... Define h

Theorem stated in lecture XIX:

Theorem 4.25 THE SOFTWARE PRINCIPLE

The function

$$g(v,w) := \begin{cases} f_{M,k}(\vec{w}) & \text{if } v = \text{code}(M),\ w = \text{code}(\vec{w}) \text{ with length } k \\ \uparrow & \text{o/w} \end{cases}$$

is computable.

This means that there is a UNIVERSAL REGISTER MACHINE U s.t.

$$f_{U,2}(v,w) = g(v,w),$$

i.e, a RM that gets $v = \text{code}(M)$ as input, regarding it as SOFTWARE and performs the computation of M.

⟶ Proof in Lecture XX.

Proof. $f : w,v \longmapsto \text{code}(\text{CCM}, \vec{w}, \#v))$ if $\text{code}(M) = w$, $\text{code}(\vec{w}) = v$

was computable by L 4.23.

Start by checking whether w is a code for RM
v is a code for a k-tuple of words.

If not: $\uparrow$.

Write $f' : w,v \longmapsto \begin{cases} a & \text{if the state of } f(w,v) \text{ is } q_H \\ \varepsilon & \text{o/w.} \end{cases}$

Minimise f' by MINIMISATION and obtain corecputable

$$h(w,v) := \text{least } v \text{ s.t. } f(w,v,v) \text{ is in state } q_H \text{ (if it exists)}$$

If $h(w,v)\uparrow$, then $g(w,v)\uparrow$, so searching for $h(w,v)$ gives the desired value.

If $h(w,v)\downarrow$, then consider

$$C(M, \vec{w}, \#h(w,v))$$

and find its code for the 0th register. Write this into the 0th register and halt.

q.e.d.

REMARKS

(1) A RM that computes g is called a universal RM.

(2) REMARKABLE.

U has a finite number of registers used & states, but can mimic the behaviour (with appr. input) of RM of arbitrary size.

(3) This allows us to streamline notation

universal machine

$$\forall v \in W \quad f_{v,k}(\vec{w}) := f_{U,2}(v, \text{code}(\vec{w}))$$

$$= f_{M,k}(\vec{w}) \text{ if } \text{code}(M) = v$$

Similarly, $W_v := \text{dom}(f_{v,1})$

So, $\{W_v : v \in W\}$ is the set of all computably enum. sets.

Theorem (s-m-n Theorem) ← Name derives from the notation used when the theorem was published.

PARAMETER THM

Suppose $g : W^{k+1} \dashrightarrow W$ partial computable function. Then there is a total computable $h : W \longrightarrow W$ s.t.

$$f_{h(v),k}(\vec{w}) = g(\vec{w}, v)$$

one variable pulled into the index

This process is called

CURRYING

after H. Curry.

Haskell CURRY
1900–1982

Proof. We want

$$f_{k(v),k}(\vec{w}) = g(\vec{w}, v).$$

Think of $g_v(\vec{w}) := g(\vec{w}, v)$.

This is computable, so there is some M^* s.t.

$$f_{M^*,k}(\vec{w}) = g_v(\vec{w}).$$

The s-m-n theorem tells us more: there is a computable k that finds M^*.

(1) $\vec{w} \longmapsto (\vec{w}, v)$ is performed by RM call it M_v.

This is just the RM that writes v into register k.

So there is a computable fn

$$v \longmapsto \text{code}(M_v).$$

(2) Since g is computable, & have RM M s.t.

$$f_{M,k+1} = g.$$

This means that g_v is computed by the RMs M_v and M in that order.

(3) We observed in §4.2 that the concatenation of two RM produces a concrete RM that we defined in the proof. Write $M \circ M_v$ for that concatenated RM.

Thus, the function

$$v \longmapsto \text{code}(M \circ M_v)$$

is total and computable.

Call this h and claim that

$$f_{h(v),k}(\vec{w}) = g(\vec{w}, v).$$

$$f_{k(v),k}(\vec{w}) = f_{\mathrm{code}(M \circ M_v),k}(\vec{w})$$
$$= f_{M \circ M_v,k}(\vec{w})$$
$$= g_v(\vec{w})$$
$$= g(\vec{w}, v).$$ q.e.d.

---

§ 4.8 Computably enumerable sets.

$$K_0 := \{(w,v) ; f_{w,1}(v)\downarrow\}$$
$$K := \{w ; f_{w,1}(w)\downarrow\}$$

HALTING PROBLEM

Theorem 4.27 Both K_0 and K are computably enumerable.

Proof. For this, it's enough to show that they are $\mathrm{dom}(f)$ for some computable f.

- By SOFTWARE PRINCIPLE $f_{v,2}(w,v) = f_{w,1}(v)$.
  $f_{v,2}$ computable with $\mathrm{dom}\ K_0$

For K, observe that

$$w \longmapsto (w,w)$$

is performed by a RM. [Copy reg. 0 to reg 1 & halt].

But then

$$w \longmapsto (w,w) \longmapsto f_{w,2}(w,w) = f_{w,1}(w)$$

is computable and its domain is K.

q.e.d.

Theorem 4.8 Neither K nor K_0 are computable.

Proof. Let's do it for K_0. [K in green]

Assume towards contradiction that K_0 is computable:

$$\chi_{K_0}(w,v) = \begin{cases} a & \text{if } (w,v) \in K_0 \\ \varepsilon & \text{o/w} \end{cases}$$

is computable. ($\chi_K(w) = a$)

$$f(w) := \begin{cases} \uparrow & \text{if } \chi_{K_0}(w,w) = a \\ \varepsilon & \text{if } \chi_{K_0}(w,w) = \varepsilon \end{cases}$$

($\chi_K(w) = \varepsilon$)

This is computable, so there is some $d \in W$ s.t. $f_{d,1} = f$.

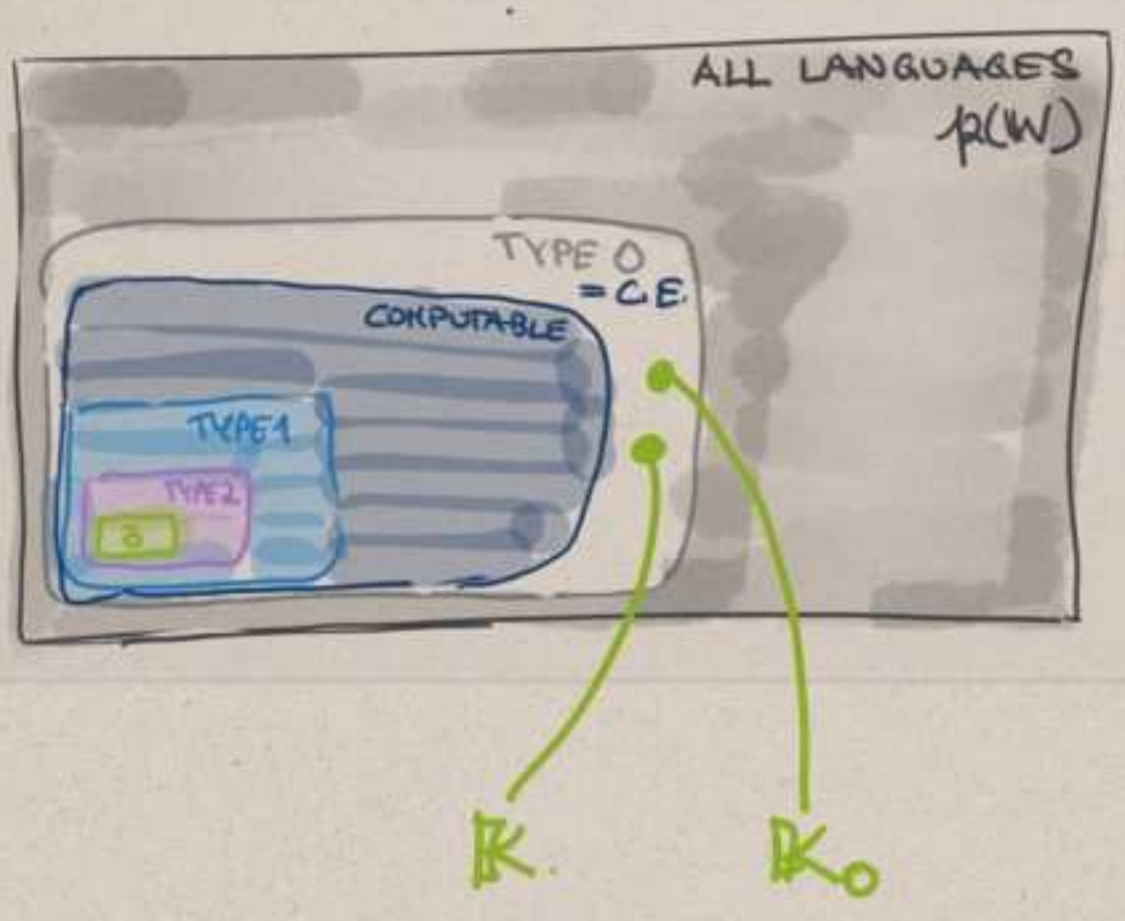
"diagonal"

$f(d)\downarrow \iff f_{d,1}(d)\downarrow \iff (d,d) \in K_0 \iff \chi_{K_0}(d,d) = a$ ($\chi_K(d) = a$)

$d \in K \iff f(d)\uparrow$. CONTRADICTION!

q.e.d.

From Lecture XVIII:

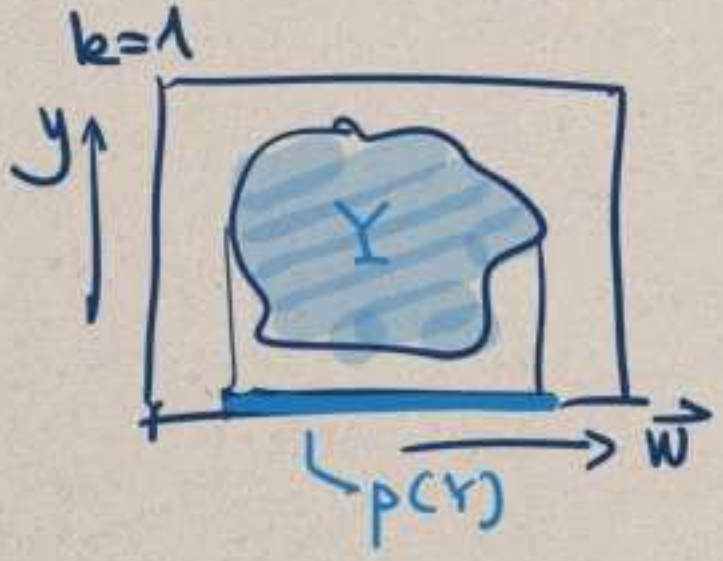


Goal Understand c.e. = Type 0.

For this, we need deeper understanding of c.e. sets.

Def. $X \subseteq W^k$ is called Σ_1 if there is a computable $Y \subseteq W^{k+1}$ s.t.

$$\vec{w} \in X \iff \exists y \ (\vec{w}, y) \in Y.$$



We say

$$X = p(Y) = \{\vec{w} ; \exists y \ (\vec{w}, y) \in Y\}$$

is the projection of Y.